

RCDs and Security Usability

Refactored Cognitive Dimensions
and Security Usability

Luke Church
luke@church.name



Agenda

- What and Why?
- RCDs in a Nutshell
- Modelling existing problems
- Usability Warfare
- Limitations and further work
- Discussion



What and Why?

- Yesterday we solved code level security problems
- But today we still have security problems
- Usability problems are harder
 - End users, not experts
 - Subtle exploits on people, not programs => Harder to reason about

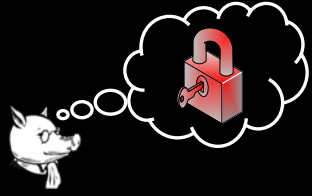


'Usability seems hard, how do you prove theorems about it?'

Approximate: Ross Anderson

'You're the only person that I know who puts Security and Usability in the same sentence'

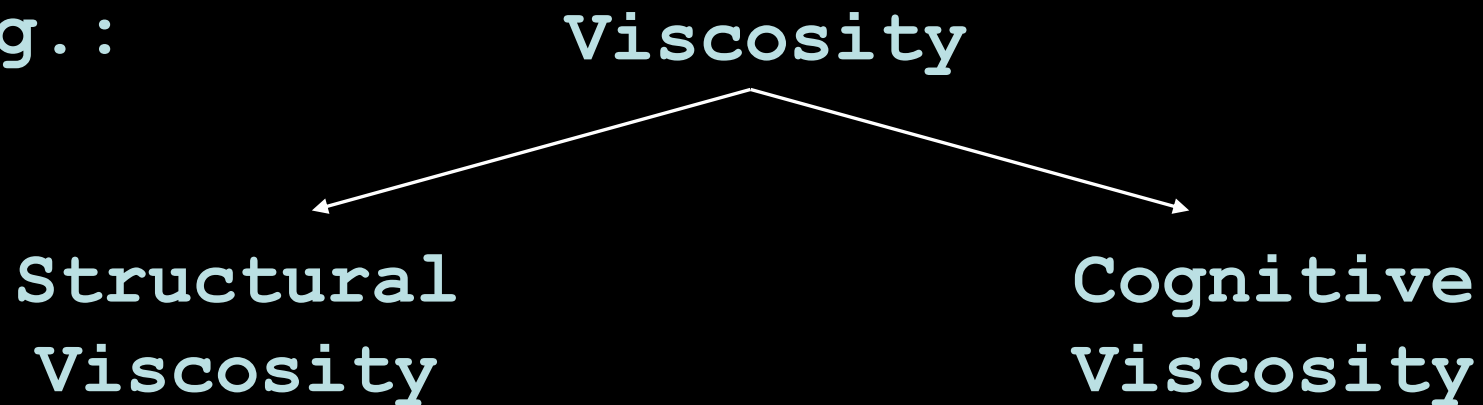
Thomas Green

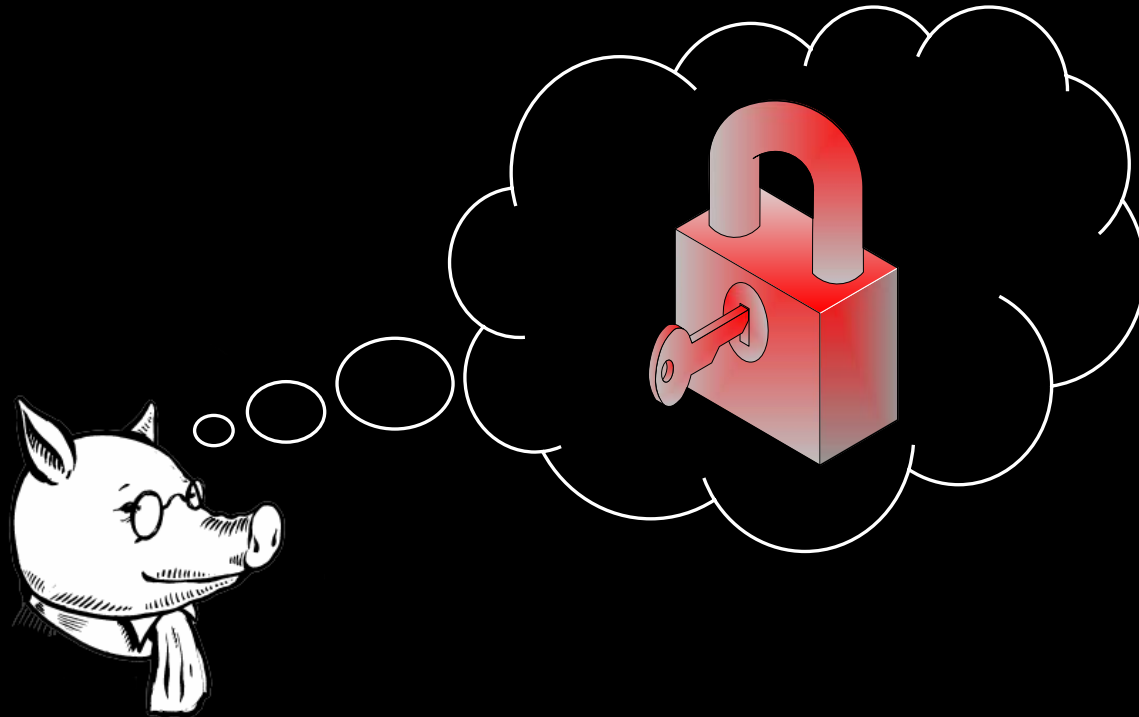


RCDs in a Nutshell

- The CDs Framework mixed information structure and cognitive concerns
 - Refactored CDs pulls them apart
 - Adds clarity required for this model

e.g.:





PART 1: Modelling

Considering existing
problems from an RCDs
perspective



Visibility

- Visibility

- How do you ask a Windows XP Pro machine about its security state?
- [Some] Information is easy to acquire, but users don't know they need to
- Windows Security Panel
 - But the complexity of the settings is much higher than this
- Ironically, the declarative programming paradigm has given low *cognitive visibility*!

Demo

Find the shared folders



Visibility Modifiers

- Tool based design manoeuvres
 - Microsoft Security Centre
 - Increases *Cognitive Visibility* and *Structural Visibility*
 - But provides a dangerous simplification
 - Microsoft Baseline Security Analyzer (MBSA)
 - Increases *Structural Visibility*
 - But inverts the state space, it has to detect 'dangerous configurations'

Demo

Security Center and MBSA



Viscosity

- Security == Viscosity Management
 - Minimise for user, maximise for attacker
- ***Viscosity Avoidance***
 - Users will minimise viscosity
 - If a security system offers too much interference, it will be disabled
 - Sudo/Runas are tradeoff tools



Discriminability

- Phishing attacks

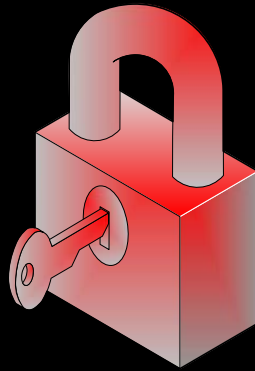
Demo

Barclays email



Further Discriminability

<code>localhost</code>	<code>localhost</code>
<code>006C, 006F, 0063, 0061, 006C, 0068, 006F, 0073, 0074</code>	<code>006C, 03BF, 0063, 0061, 006C, 0068, 006F, 0073, 0074</code>
<code>172.0.0.1</code>	<code>DNS Lookup</code>



PART 2: Usability Warfare

Attack and Defence in
Usability Space



Viscosity Manipulation Attack

- **Observation:** If a system is causing too much viscosity, it will be disabled
 - So the attack doesn't need to break your firewall, only make it so annoying that you turn it off
 - Or so you start ignoring its warnings
- **Viscosity Manipulation is often easy**



Hidden Dependency Optimisation

- **CONTENT REMOVED** due to NDA



RCDs analysis

- Analysis

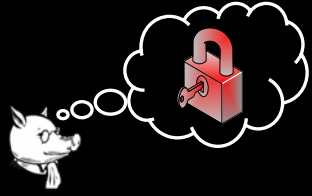
- Build a search system: Information issues

- The rest are cognitive

- How can we manipulate

- The search system?

- The cognitive properties?



...

- Further CONTENT REMOVED due to NDA



Usability Effects

- **Viscosity Duality**
 - Intended mode of operation -> Low viscosity
 - Different mode -> Very high viscosity
- Through a manipulation of the *Hidden Dependencies* the *Structural Viscosity* of the system has been vastly increased



Issues

- 'Defending the valueless against the disinterested'
 - But all security is
 - Make it hard enough
- RCDs clarifies the design manoeuvres
- Usability can provide security wins and loses, where information theory can't
 - It doesn't matter if your enemy can win, as long as it's so hard they don't bother trying!



Problems

- RCDs are not enough to build the model
 - Visual Effects
 - Temporarily
 - 'Human Issues'
 - Acceptability
 - Motivation



Conclusion

- A lot of Security Usability issues are cognitive, not technical
- An attacker only needs to be able to manipulate the cognitive properties of your system to win
- RCDs gives us a perspective on information security that information theory cannot

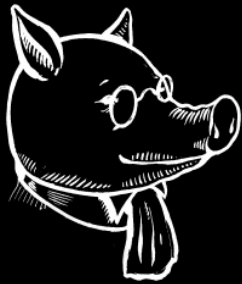


Where to go from here?

- Certain lack of coherence
 - But the approach seems to be interesting
- Need to look at a larger dataset to understand subtle usability effects
- Even a heavily incomplete model seems useful



Discussion?



luke@church.name